



MizoPol

Tools for parsimonious characterization of industrial time-series

equipment sensors time-series



- Anomaly detection
- Virtual sensors
- Digital twins
- Dynamic of aging
- Invariants design
- Synthetic fault generation

MizoPol's keywords

- What is a Parsimonious Characterization?
- Advantages of parsimony
- Parsimony in Industrial context
- Explainability
- Scanning the graph of coupling

MizoPol in a glance

- Plars, G2sys, PwPol, Xpwpol & Rlars

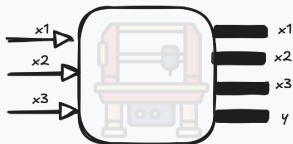
Case Studies

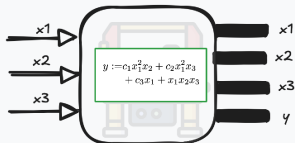
- Soon-to-fail alarm for turbofans
- Anomaly detection in manipulator robots
- Events Detection During Anaesthesia
- Explicit parsimonious models in robotics
- Sparsity & generalization to unseen contexts
- Fault detection in the presence of unknown multiple context

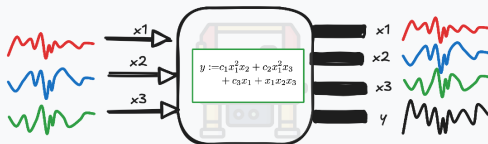
Deployment | API

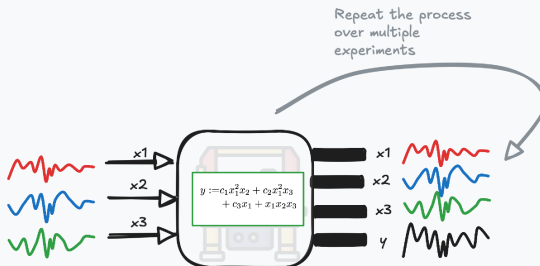
MizoPol's keywords

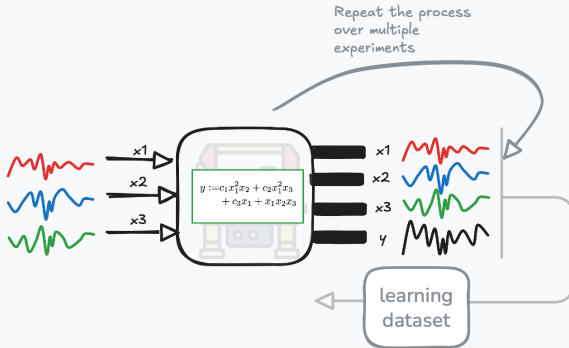
What is a **parsimonious characterization**

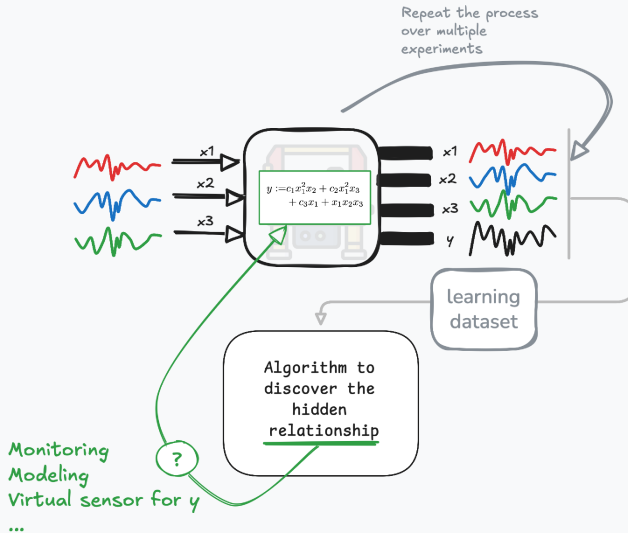












Training context

$$\mathbb{X}_{\text{train}} := [-1, 1]^3$$

Test context

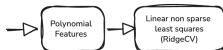
$$\mathbb{X}_{\text{test}} := [-5, 5]^3$$

Compared solutions

3 Deep NN

(8,8,2), (32, 16, 4), (128, 128, 32)

Non sparse Wiener structure

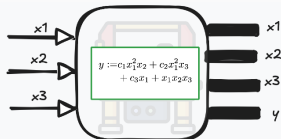


Mizopol | plars
(Sparse polynomial)



The regression error e_y

$$e_y := \frac{\text{percentile}(|y - \hat{y}|, 95)}{\text{percentile}(|y|, 50)}$$



Training context

$$\mathbb{X}_{\text{train}} := [-1, 1]^3$$

Test context

$$\mathbb{X}_{\text{test}} := [-5, 5]^3$$

Compared solutions

3 Deep NN
(8,8,2), (32, 16, 4), (128, 128, 32)

Non sparse Wiener structure

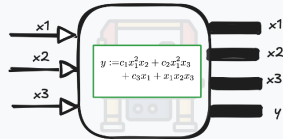


Mizopol | plars
(Sparse polynomial)

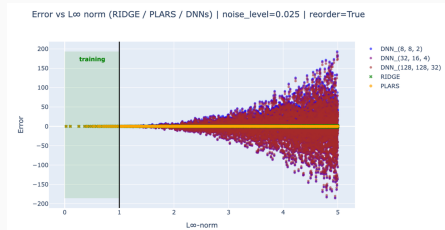


The regression error e_y

$$e_y := \frac{\text{percentile}(|y - \hat{y}|, 95)}{\text{percentile}(|y|, 50)}$$



Normalized regression errors



Training context

$$\mathbb{X}_{\text{train}} := [-1, 1]^3$$

Test context

$$\mathbb{X}_{\text{test}} := [-5, 5]^3$$

Compared solutions

3 Deep NN
(8,8,2), (32, 16, 4), (128, 128, 32)

Non sparse Wiener structure

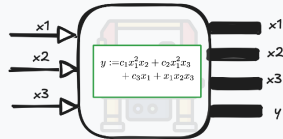


Mizopol | plars
(Sparse polynomial)

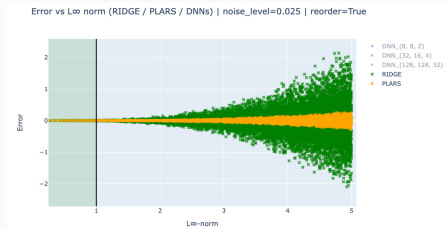


The regression error e_y

$$e_y := \frac{\text{percentile}(|y - \hat{y}|, 95)}{\text{percentile}(|y|, 50)}$$



Normalized regression errors



Training context

$$\mathbb{X}_{\text{train}} := [-1, 1]^3$$

Test context

$$\mathbb{X}_{\text{test}} := [-5, 5]^3$$

Compared solutions

3 Deep NN
(8,8,2), (32, 16, 4), (128, 128, 32)

Non sparse Wiener structure

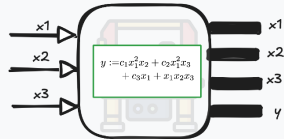


Mizopol | plars
(Sparse polynomial)

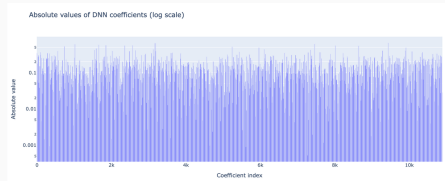


The regression error e_y

$$e_y := \frac{\text{percentile}(|y - \hat{y}|, 95)}{\text{percentile}(|y|, 50)}$$



Number of non zero coefficients in the solution



Test context

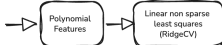
$$\mathbf{X}_{\text{train}} := [-1, 1]^3$$

$$\mathbb{X}_{\text{test}} := [-5, 5]^3$$

Compared solutions

3 Deep NN
(8,8,2), (32, 16, 4), (128, 128, 32)

Non sparse Wiener structure

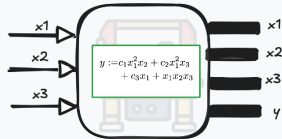


Mizopol | plars
(Sparse polynomial)



The regression error e_y

$$e_y := \frac{\text{percentile}(|y - \hat{y}|, 95)}{\text{percentile}(|y|, 50)}$$



Number of non zero coefficients in the solution



Training context

$$\mathbb{X}_{\text{train}} := [-1, 1]^3$$

Test context

$$\mathbb{X}_{\text{test}} := [-5, 5]^3$$

Compared solutions

3 Deep NN
(8,8,2), (32, 16, 4), (128, 128, 32)

Non sparse Wiener structure

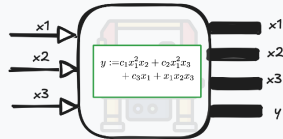


Mizopol | plars
(Sparse polynomial)



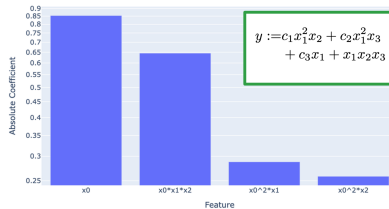
The regression error e_y

$$e_y := \frac{\text{percentile}(|y - \hat{y}|, 95)}{\text{percentile}(|y|, 50)}$$

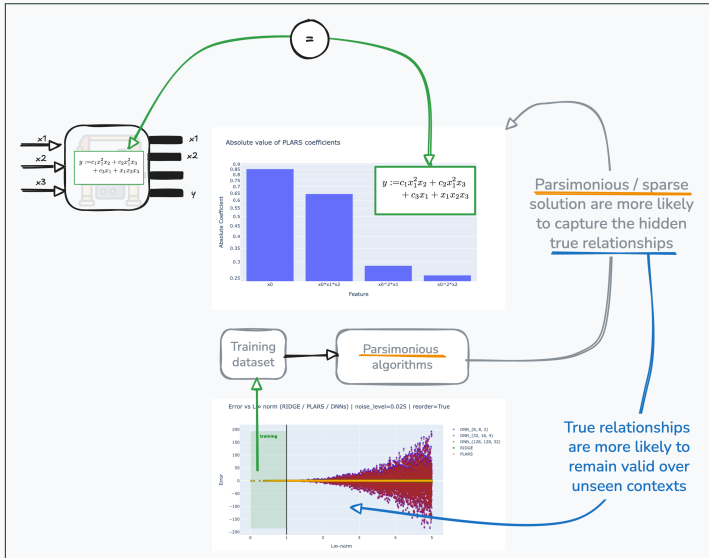


Number of non zero coefficients in the solution

Absolute value of PLARS coefficients

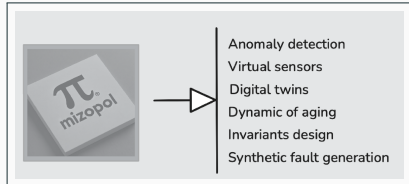


To summarize



Advantages of **parsimonious** characterizations

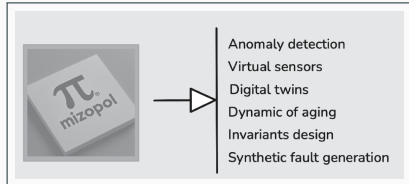
Advantages of parsimony



Advantages of parsimony



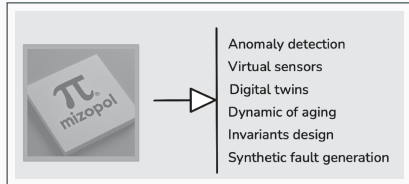
✓ **Robustness to unseen excursions**



Advantages of parsimony



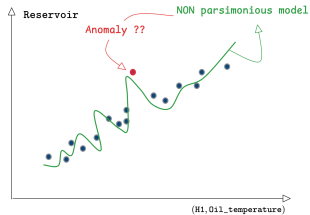
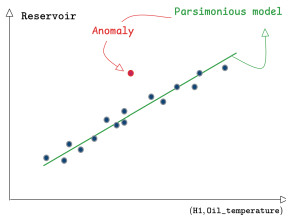
- ✓ Robustness to unseen excursions
- ✓ **Reduction** in False Alarm rate



Advantages of parsimony



- ✓ Robustness to unseen excursions
- ✓ Reduction in False Alarm rate
- ✓ **Higher** detection rate



Advantages of parsimony



- ✓ Robustness to unseen excursions
- ✓ Reduction in False Alarm rate
- ✓ Higher detection rate
- ✓ **Data efficient** processing



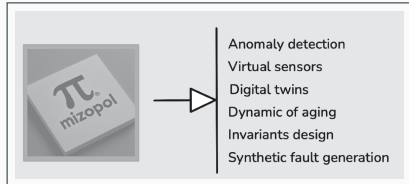
→

- Anomaly detection
- Virtual sensors
- Digital twins
- Dynamic of aging
- Invariants design
- Synthetic fault generation

Advantages of parsimony



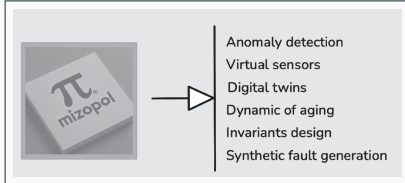
- ✓ Robustness to unseen excursions
- ✓ Reduction in False Alarm rate
- ✓ Higher detection rate
- ✓ Data efficient processing
- ✓ **Reliable** induced digital-twins



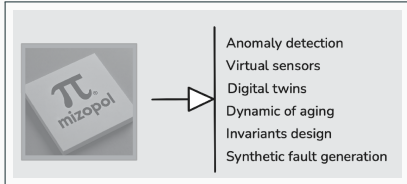
Advantages of parsimony



- ✓ Robustness to unseen excursions
- ✓ Reduction in False Alarm rate
- ✓ Higher detection rate
- ✓ Data efficient processing
- ✓ Reliable induced digital-twins
- ✓ **Reliable induced virtual sensors**



Advantages of parsimony

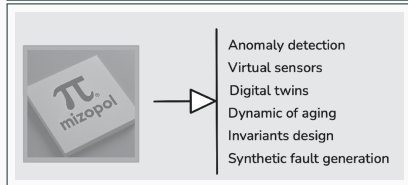


- ✓ **Explainability** & default localization (see later)

Advantages of parsimony



- ✓ Robustness to unseen excursions
- ✓ Reduction in False Alarm rate
- ✓ Higher detection rate
- ✓ Data efficient processing
- ✓ Reliable induced digital-twins
- ✓ Reliable induced virtual sensors



- ✓ Explainability & default localization (see later)
- ✓ Building **Graph** of coupling between sensors (see later)

Advantages of parsimony



- ✓ **Robustness** to unseen excursions
- ✓ **Reduction** in False Alarm rate
- ✓ **Higher** detection rate
- ✓ **Data efficient** processing
- ✓ **Reliable** induced digital-twins
- ✓ **Reliable** induced virtual sensors

- ✓ **Explainability** & default localization ([see later](#))
- ✓ Building **Graph** of coupling between sensors ([see later](#))



Anomaly detection
Virtual sensors
Digital twins
Dynamic of aging
Invariants design
Synthetic fault generation

Parsimony in Industrial Context

Physical laws governing equipments **are parsimonious**



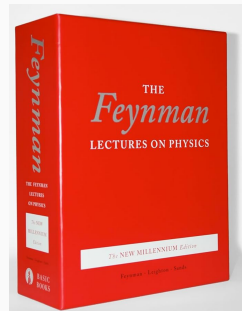
$$\begin{aligned} J \dot{\omega} &= T_m - \frac{\rho g Q H}{\omega} \\ H &= H_0 - b Q - a Q^2 \\ Q &= k_Q \omega - k_\ell \rho g H \\ T_m &= k_t u - d \omega \end{aligned}$$

9 parameters

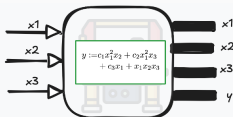


$$\begin{aligned} M(q, p) \ddot{q} &= C(q, \dot{q}, p) \dot{q} + \\ &+ G(q, p) + \\ &+ T(q, \dot{q}, p) i \end{aligned}$$

≤ 84 parameters



Centuries of evidences



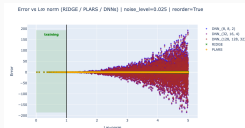
These physical laws may not be necessarily fully known. Nevertheless, they are undoubtedly **parsimonious**.

Exhaustive covering of all possible excursions is **impossible**



The possible features excursions depend on:

- ✓ Set-points
- ✓ Trajectories
- ✓ Load dynamics
- ✓ Type of task
- ✓ ...



It is rarely possible to include all possible excursion in the training dataset. **Sparsity-induced robustness** is required for the characterization to hold over **unseen excursions**.

Why Parsimony Enhances Explainability?

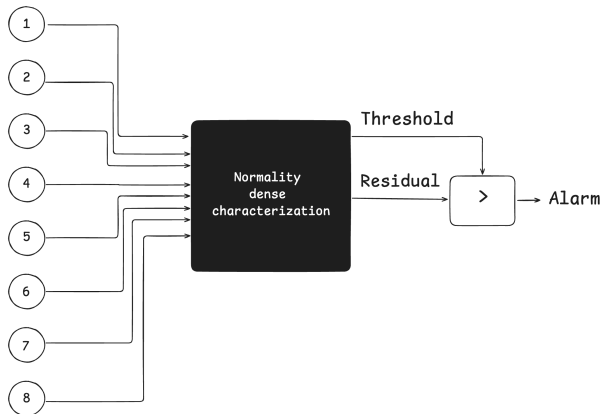
- ✓ **Robustness** to unseen excursions
- ✓ **Reduction** in False Alarm rate
- ✓ **Higher** detection rate
- ✓ **Data efficient** processing
- ✓ **Reliable** induced digital-twins
- ✓ **Reliable** induced virtual sensors

- ✓ **Explainability** & default localization ([see later](#))
- ✓ Building **Graph** of coupling between sensors ([see later](#))

Back to claimed advantages!

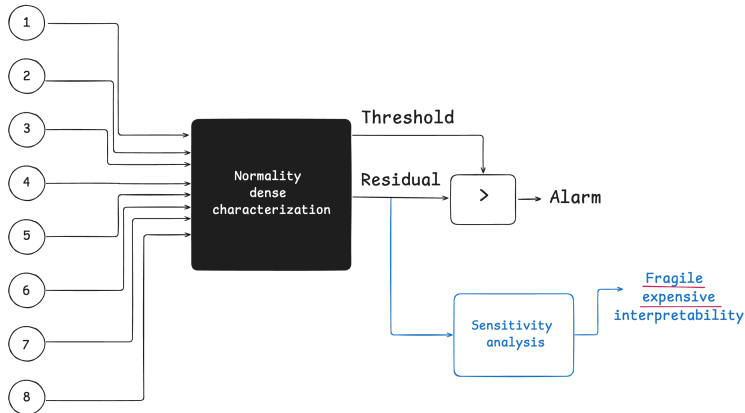
Conventional explainability as a **hard inverse problem**

Sensors



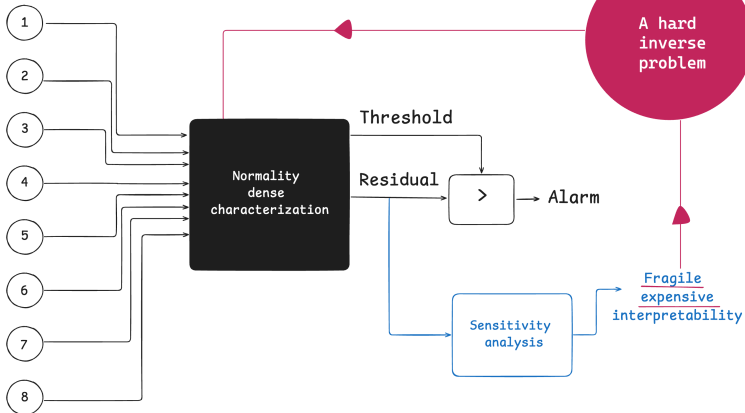
Conventional explainability as a **hard inverse problem**

Sensors

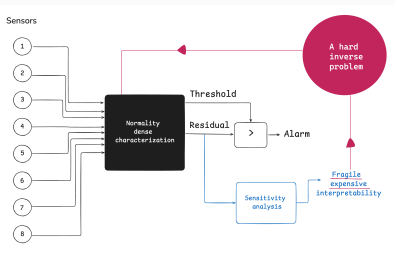


Conventional explainability as a **hard inverse problem**

Sensors



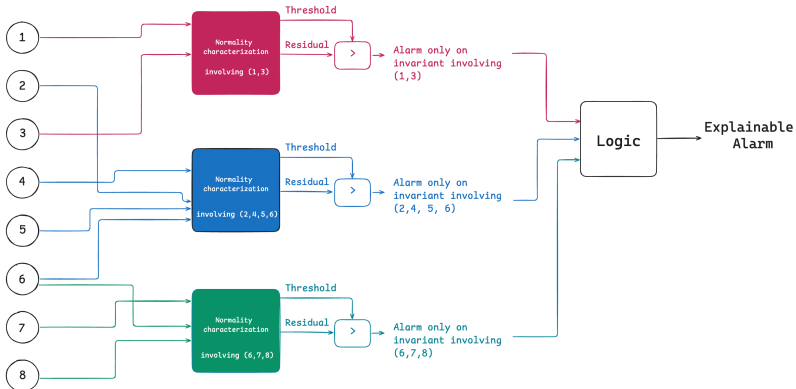
Conventional explainability as a **hard inverse problem**



- ★ All coefficients are generically non zero
- ★ Some are high for specific configurations
- ★ They get compensated for in others
- ★ A posteriori **difficult** search for sparsity

MizoPol sparsity-induced explainability

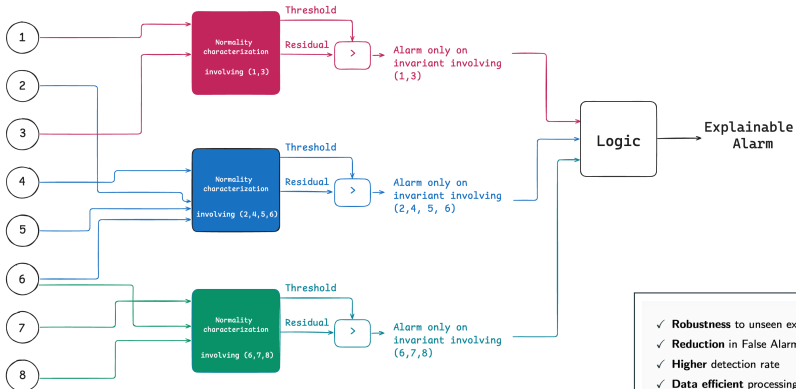
Sensors



Structural (a priori) explainability

MizoPol sparsity-induced explainability

Sensors



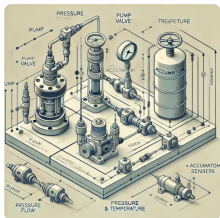
Structural (a priori) explainability

- ✓ **Robustness** to unseen excursions
- ✓ **Reduction** in False Alarm rate
- ✓ **Higher** detection rate
- ✓ **Data efficient** processing
- ✓ **Reliable** induced digital-twins
- ✓ **Reliable** induced virtual sensors

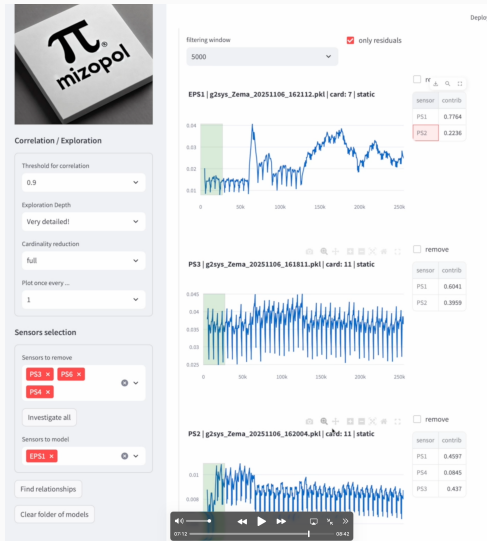
✓ **Explainability & default localization** (see later)

✓ Building **Graph** of coupling between sensors (see later)

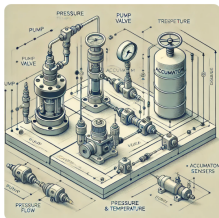
MizoPol sparsity-induced explainability: An example



Detecting invariants and anomalies in a hydraulic systems. (The above AI-generated photo is used for illustrative purpose and is not directly related to the dataset.)



MizoPol sparsity-induced explainability: An example



Detecting invariants and anomalies in a hydraulic systems. (The above AI-generated photo is used for illustrative purpose and is not directly related to the dataset.)



Problem:

PS1, PS2 or EPS1?



Correlation / Exploration

Threshold for correlation
0.9

Exploration Depth
Very detailed!

Cardinality reduction
full

Plot once every ...
1

Sensors selection

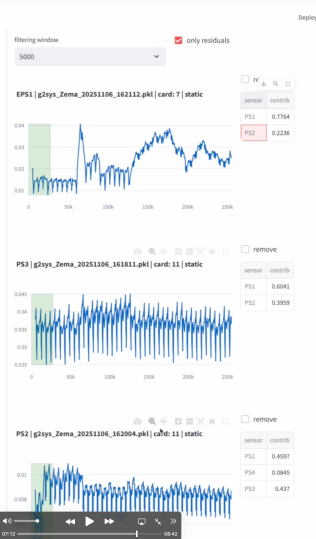
Sensors to remove
PS3 x PS6 x PS4 x

Investigate all

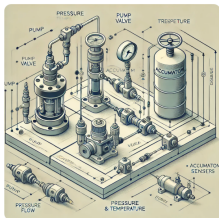
Sensors to model
EPS1 x

Find relationships

Clear folder of models



MizoPol sparsity-induced explainability: An example



Detecting invariants and anomalies in a hydraulic systems. (The above AI-generated photo is used for illustrative purpose and is not directly related to the dataset.)



No Problem in PS1, PS2

→ EPS1



Correlation / Exploration

Threshold for correlation
0.9

Exploration Depth
Very detailed!

Cardinality reduction
full

Plot once every ...
1

Sensors selection

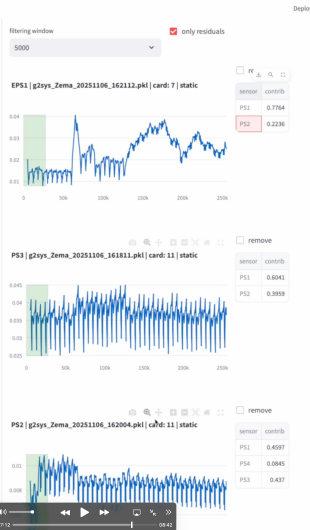
Sensors to remove
PS3 × PS6 × PS4 ×

Investigate all

Sensors to model
EPS1 ×

Find relationships

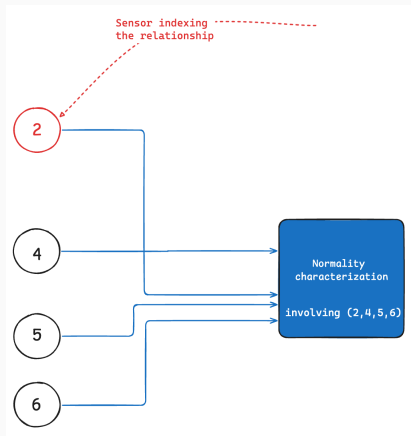
Clear folder of models

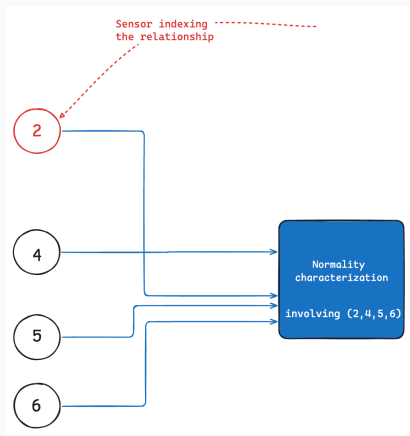


Scanning The Graph of **Sensors Coupling**

- ✓ **Robustness** to unseen excursions
 - ✓ **Reduction** in False Alarm rate
 - ✓ **Higher** detection rate
 - ✓ **Data efficient** processing
 - ✓ **Reliable** induced digital-twins
 - ✓ **Reliable** induced virtual sensors
-
- ✓ **Explainability** & default localization [\(see later\)](#)
 - ✓ Building **Graph** of coupling between sensors [\(see later\)](#)

MizoPol Building sensors graph





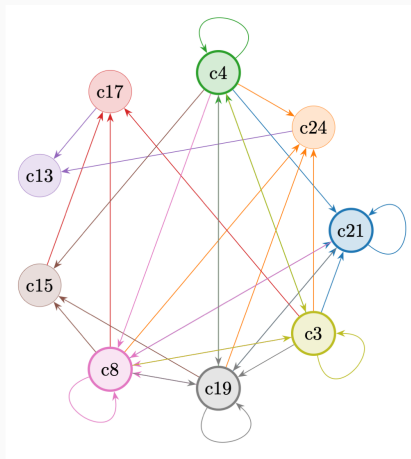
Two classes of relationships

Non recursive

$$s_2 = P(s_4, s_5, s_6, \text{delayed instances})$$

Recursive

$$\Delta s_2 = P(s_2, s_4, s_5, s_6, \text{delayed instances})$$



Graph of sensors for the NASA's Turbo Fans Dataset

Two classes of relationships

Non recursive

$$s_2 = P(s_4, s_5, s_6, \text{delayed instances})$$

Recursive

$$\Delta s_2 = P(s_2, s_4, s_5, s_6, \text{delayed instances})$$

- ✓ **Robustness** to unseen excursions
- ✓ **Reduction** in False Alarm rate
- ✓ **Higher** detection rate
- ✓ **Data efficient** processing
- ✓ **Reliable** induced digital-twins
- ✓ **Reliable** induced virtual sensors

- ✓ **Explainability** & default localization (see later)

- ✓ **Building Graph** of coupling between sensors (see later)

MizoPol in a glance

Plars

Explicit polynomial

G2sys

St./dyn. ExPol / delays

PwPol

Implicit piece-wise pol

XpwPol

Explicit piece-wise pol.

Rlars

Rational/roots of pol.

Quick Overview of the MizoPol components



Plars Explicit polynomial	G2sys St./dyn. ExPol / delays	PwPol Implicit piece-wise pol	XpwPol Explicit piece-wise pol.	Rlars Rational/roots of pol.
-------------------------------------	---	---	---	--

Plars

Explicit single multivariate polynomials.

① Problem statement

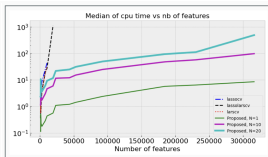
Given

- a matrix of features X and
- a label vector y ,

the **plars** module enables to fit a **sparse** polynomial P s.t. $y \approx P(x)$, namely:

$$y \approx P(x) = \sum_{i=1}^{n_c} c_i \phi_i(x) \quad \text{where} \quad \phi_i(x) = \prod_{j=1}^n x_j^{p_{ij}} \quad (1)$$

By **sparse** it is meant that the polynomial P is computed in a way that enhances the fact that it involves a small number n_c of monomials with non vanishing coefficients¹ c_i .



n	5	5	5	10	10	10	20	20	20	50	50	50
deg	2	4	5	2	4	5	2	4	5	2	4	5
nc	21	126	252	66	1001	3003	231	10626	53130	1326	316251	3478761

Figure 1: Number of candidate monomials for different values of the pair $(n, d = \text{deg})$

Plars	G2sys	PwPol	XpWPol	Rlars
Explicit polynomial	St./dyn. ExPol / delays	Implicit piece-wise pol	Explicit piece-wise pol.	Rational/roots of pol.

i Static

$$s_i(k) = F_i(\{s_j(k - \sigma \times d_i)\}_{j \neq i, \sigma \in \{0, \dots, m_i\}})$$

→ s_i is a static function of the other sensors and their past values up to $m_i \times d_i$.

i Dynamic

$$[\Delta^{(m_i)} s_i](k) = F_i \left(\begin{array}{c} s_i(k - \sigma \times d_i) \\ \{s_j(k - \sigma \times d_i)\}_{j \neq i, \sigma \in \{0, \dots, m_i\}} \end{array} \right)$$

→ The increment $[\Delta^{(m_i)} s_i](k)$ represents the discrete version, at instant k , of the m_i derivatives of s_i .

G2sys

Explicit single multivariate polynomials representing static or dynamic relationships with delays.

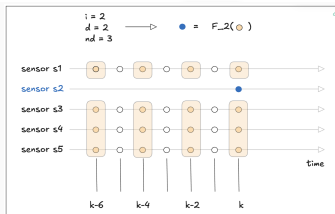


Figure 1: Illustration of the logic of a static relationship indexed by sensor 2 with $n_d = 3$, $d = 2$.

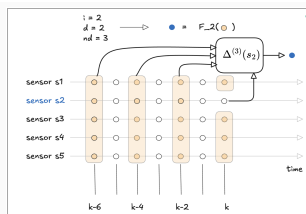


Figure 2: Illustration of the dynamic relationship associated to $i = 2$, $d = 2$ and $n_d = 3$.

Plars Explicit polynomial	G2sys St./dyn. ExPol / delays	PwPol Implicit piece-wise pol	XpwPol Explicit piece-wise pol.	Rlars Rational/roots of pol.
-------------------------------------	---	---	---	--

PwPol

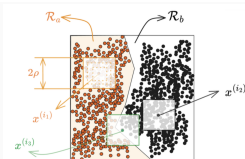
Implicit
piece-wise
multivariate
polynomials
generating
residuals.

i Problem statement

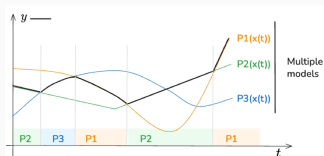
Given a set of sensors $s_j, j = 1, \dots, n_s$ and a target sensor s_i , find a set of n_r polynomials $P_\sigma, \sigma = 1, \dots, n_r$, such that the following residual:

$$e_i = \min_{\sigma=1}^{n_r} |s_i - P_\sigma(\{s_j\}_{j \neq i})| \quad (1)$$

is small.



Schematic view of a piece-wise defined relationship with two regions R_a and R_b and the random search hypercubes used in the **pwpol** module.



Plars Explicit polynomial	G2sys St./dyn. ExPol / delays	PwPol Implicit piece-wise pol	Xpwpol Explicit piece-wise pol.	Rlars Rational/roots of pol.
-------------------------------------	---	---	---	--

The explicit relationship provided by the **xpwpol** module takes the following form:

$$\hat{y} = \frac{1}{n_v} \sum_{\kappa=1}^{n_v} P_{\sigma(x|\kappa)}(x) \quad \text{where} \quad \sigma(x|\kappa) \in \{1, \dots, n_r\}$$

Xpwpol

Explicit
piece-wise
multivariate
polynomials.

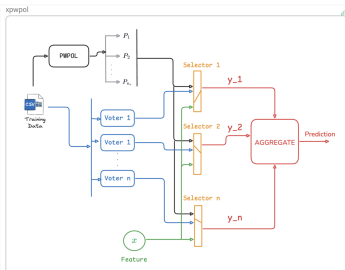


Figure 1: Schematic view of the principle of the xpwpol module. The number n of voters is precisely the integer n_v used in Equation 1. The aggregation box performs the average of the values predicted by the different voters although any different kind of aggregation can be implemented.

α	Error	α	Error	α	Error
50%	0.038135	50%	0.187046	50%	0.275005
80%	0.079292	80%	0.343992	80%	0.501445
90%	0.106521	90%	0.457063	90%	0.716726
95%	0.137383	95%	0.571375	95%	0.966963
98%	0.218389	98%	0.726146	98%	1.348832
99%	0.308927	99%	0.852698	99%	1.657691
100%	1.465261	100%	2.144246	100%	6.015367

pwpol residuals (Implicit)

xpwpol residuals (Explicit)

plars residuals (Explicit)

Plars

Explicit polynomial

G2sys

St./dyn. ExPol / delays

PwPol

Implicit piece-wise pol

XpwPol

Explicit piece-wise pol.

Rlars

Rational/roots of pol.

Rlars

Label y is a root of x -dependent multivariate-polynomials (Implicit)

Special Case: Explicit rational-polynomial expressions.

Given,

- a vector of features x and
- a label y ,

rlars looks for a set of multi-variate polynomials:

$$c_0(x), \dots, c_r(x)$$

such that y is a root of the following scalar polyomial (in the unknown z):

$$c_r(x)z^r + \dots + c_1(x)z + c_0(x)$$

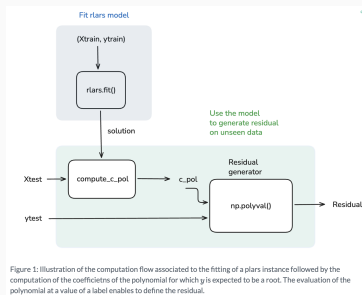


Figure 1: Illustration of the computation flow associated with the fitting of a plars instance followed by the computation of the coefficients of the polynomial for which y is expected to be a root. The evaluation of the polynomial at a value of a label enables to define the residual.

$$y = -\frac{\bar{c}_0(x)}{\bar{c}_1(x)}$$

← particular frequent case

 **MizoPol** [Home](#) [Overview](#) [Plars](#) [g2sys](#) [pwpol](#) [xwpol](#) [rlars](#) [Graph](#) [Digital twins](#) [About](#) [Mazen Alami](#)  

MizoPol

Preliminaries

Overview

Polynomials

piecewise polynomials

Parsimony: Intro

Parsimony: Example 1

Parsimony: Example 2

True/False anomalies

plars

plars: Introduction

plars: Principle

plars: Example of use

plars: GUI

plars: Robotic

g2sys

g2sys: Introduction

g2sys: Principle

g2sys: Delays matter

g2sys: Hydraulic rig

g2sys: Compressor

g2sys: Valves

g2sys: turbofan

MizoPol

A set of tools for identification, normality characterization and anomaly detection in industrial time-series.



MizoPol is a set of python modules¹ that addresses the problem of **identifying (piece-wise) polynomial relationships** between time series. This identification is detrimental in solving the following problems:

- Anomaly detection by normality characterization
- Understanding the coupling between sensors beyond linear correlations
- Identifying dynamics governing nonlinear unmodelled systems
- Creating virtual sensors
- Generating parametric subtil defaults to auto-train models
- help designing digital twins

 **Domains of application**

Achieving the above listed tasks paves the way towards addressing many important domains of interest in industry such as:

https://mazenalamir.github.io/mizopol_documentation/

Case Studies

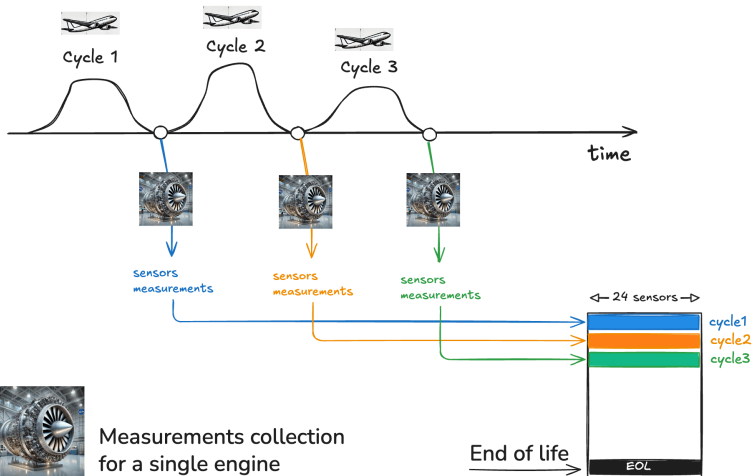


Advanced Age alarm of NASA Turbo-Fans





Data collection





Data collection





Four datasets for different contexts





Statistics of the number of cycles to EOL



Statistics of

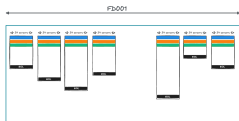
EOL

	FD001	FD002	FD003	FD004
$\text{mean}(c_{\max})$	206	206	247	245
$\text{std-dev}(c_{\max})$	46	46	86	72
$\text{min}(c_{\max})$	128	128	145	128
$\text{max}(c_{\max})$	362	378	525	543
n_{eq}	100	260	100	249

Table 2: Statistics of the number of life cycles of the engines present in the different datasets and the total number of engines in each dataset.



Problem Statement



Design an algorithm and a threshold th such that:

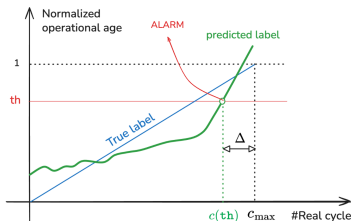
1. Δ is always > 0
2. Δ is statistically decently small

Statistics of

EOL

	FD001	FD002	FD003	FD004
$\text{mean}(c_{max})$	206	206	247	245
$\text{std-dev}(c_{max})$	46	46	86	72
$\text{min}(c_{max})$	128	128	145	128
$\text{max}(c_{max})$	362	378	525	543
n_{eq}	100	260	100	249

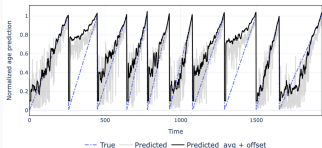
Table 2: Statistics of the number of life cycles of the engines present in the different datasets and the total number of engines in each dataset.



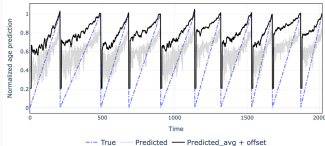


Learning on 10 engines

Normalized prediction on training data: NASA FD001, Trial 2, Config#1, avg_win 10



Normalized prediction on training data: NASA FD002, Trial 2, Config#1, avg_win 10



Sparsity of the solution

Dataset	Config	mean cardinality	std-dev cardinality
FD001	Config1	148.7	17.1
	Config2	2.1	0.2
	Config3	20.9	3.2
FD002	Config1	675.5	63.4
	Config2	5.4	3.7
	Config3	100.4	18.6
FD003	Config1	288.9	45.9
	Config2	3.4	0.6
	Config3	42.5	8.7
FD004	Config1	931.9	113.0
	Config2	7.2	3.7
	Config3	129.8	18.8

Statistics of the number of cycles before EOL

	FD001	FD002	FD003	FD004
mean(c_{max})	206	206	247	245
std-dev(c_{max})	46	46	86	72
min(c_{max})	128	128	145	128
max(c_{max})	362	378	525	543
n_{eq}	100	260	100	249

Table 2: Statistics of the number of life cycles of the engines present in the different datasets and the total number of engines in each dataset.

Config # 1 Bagging over 63 Plars explicit sparse models.

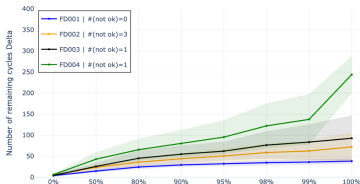
$$(\Delta^{50\%}, \Delta^{80\%} | loss)$$

Dataset	th=0.9	th=0.95	th=0.98
FD001	(26, 43 0.0%)	(19, 24 0.0%)	(9, 15 5.4%)
FD002	(46, 72 0.0%)	(24, 36 1.1%)	(14, 22 1.4%)
FD003	(42, 67 0.0%)	(26, 45 1.0%)	(18, 36 15.0%)
FD004	(69, 105 0.0%)	(43, 65 0.5%)	(31, 50 0.45%)



Statistics of Δ for $\text{th}=0.95$

Expectation and std-deviation of Percentiles | $\text{th}=0.95$



M. Alamir. On designing soon-to-fail alarm system for industrial equipments via sparse identification of ageing indicators: paradigm, algorithms and case-studies <https://cnrs.hal.science/hal-05518495>. 2026

Sparsity of the solution

Dataset	Config	mean cardinality	std-dev cardinality
FD001	Config1	148.7	17.1
	Config2	2.1	0.2
	Config3	20.9	3.2
FD002	Config1	675.5	63.4
	Config2	5.4	3.7
	Config3	100.4	18.6
FD003	Config1	288.9	45.9
	Config2	3.4	0.6
	Config3	42.5	8.7
FD004	Config1	931.9	113.0
	Config2	7.2	3.7
	Config3	129.8	18.8

Statistics of the number of cycles before EOL

	FD001	FD002	FD003	FD004
$\text{mean}(c_{\max})$	206	206	247	245
$\text{std-dev}(c_{\max})$	46	46	86	72
$\text{min}(c_{\max})$	128	128	145	128
$\text{max}(c_{\max})$	362	378	525	543
n_{eq}	100	260	100	249

Table 2: Statistics of the number of life cycles of the engines present in the different datasets and the total number of engines in each dataset.

Config # 1 Bagging over 63 Plars explicit sparse models.

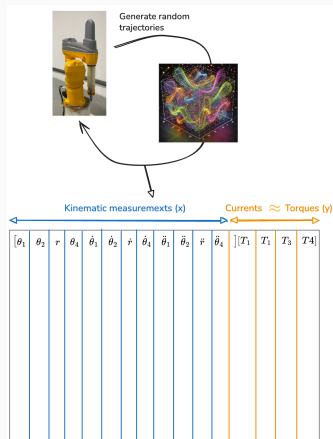
$$(\Delta^{50\%}, \Delta^{80\%} | \text{loss})$$

Dataset	$\text{th}=0.9$	$\text{th}=0.95$	$\text{th}=0.98$
FD001	(26, 43 0.0%)	(19, 24 0.0%)	(9, 15 5.4%)
FD002	(46, 72 0.0%)	(24, 36 1.1%)	(14, 22 1.4%)
FD003	(42, 67 0.0%)	(26, 45 1.0%)	(18, 36 15.0%)
FD004	(69, 105 0.0%)	(43, 65 0.5%)	(31, 50 0.45%)



Anomaly Detection in Manipulator Robots

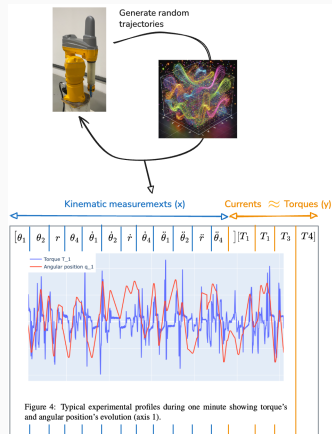
Data collection. A set of 200 experiments representing random movements along the 4 axis $\rightarrow$ **2,241,664** samples



Anomaly detector for manipulator robots



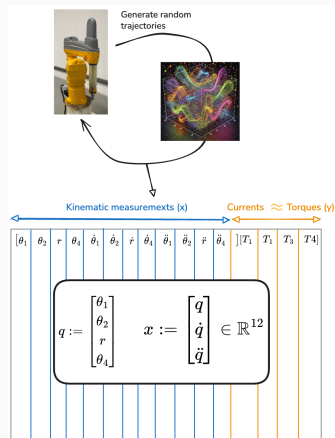
Data collection. A set of 200 experiments representing random movements along the 4 axis → **2,241,664** samples



Anomaly detector for manipulator robots

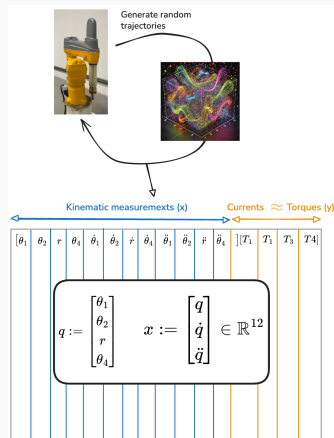
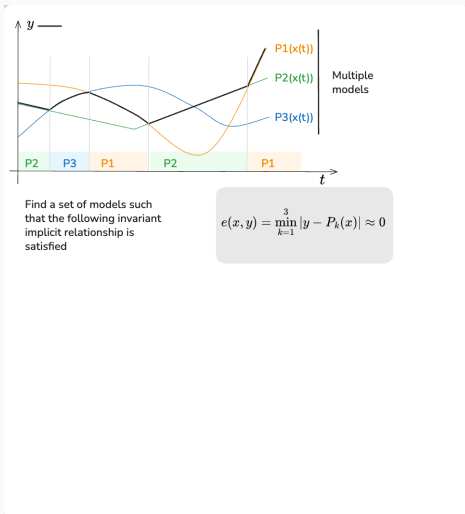


Data collection. A set of 200 experiments representing random movements along the 4 axis $\rightarrow$ **2,241,664** samples





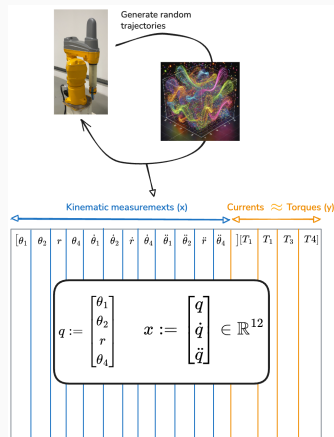
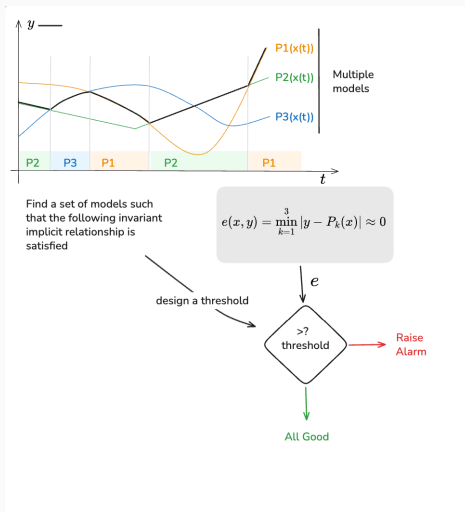
Data collection. A set of 200 experiments representing random movements along the 4 axis → **2,241,664** samples



Anomaly detector for manipulator robots



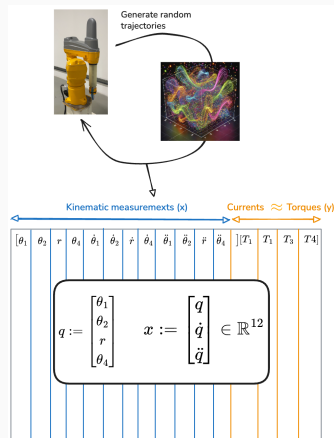
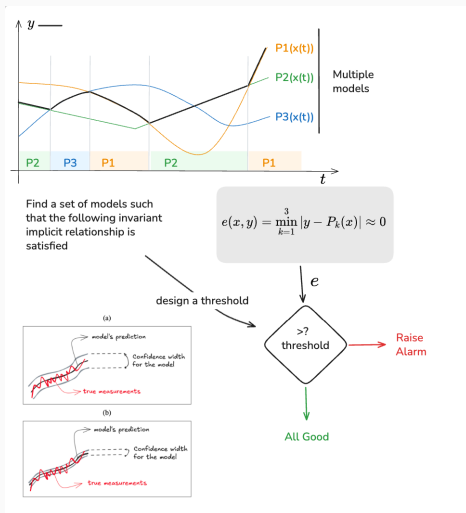
Data collection. A set of 200 experiments representing random movements along the 4 axis → **2,241,664** samples



Anomaly detector for manipulator robots



Data collection. A set of 200 experiments representing random movements along the 4 axis → **2,241,664** samples





Sparsity & CPU | parsimonious

th	#models (n_m)	#coeffs	cpu (sec)	axis
0.4	2	55	15	1
0.4	3	72	15	2
0.4	4	112	22	3
0.4	3	95	19	4
0.3	3	74	16	1
0.3	5	150	20	2
0.3	7	242	27	3
0.3	5	146	22	4
0.25	4	116	20	1
0.25	6	159	24	2
0.25	5	168	22	3
0.25	9	271	33	4
0.2	6	135	28	1
0.2	9	262	38	2
0.2	13	424	33	3
0.2	13	380	40	4
0.15	13	387	39	1
0.15	10	276	40	2
0.15	16	534	41	3
0.15	20	556	53	4
0.1	22	585	63	1
0.1	29	871	78	2
0.1	19	621	60	3
0.1	32	869	86	4

Table 1: Characteristics (number of polynomials n_m , total number #coeffs of non zero coefficients) and computation times of the identified models for precision thresholds $\text{th} \in \{0.1, 0.15, 0.2, 0.25, 0.3, 0.4\}$. (cpu times are obtained on a MacBook Pro, Apple M3 Pro, 18 Go RAM running on Sequoia 15.3.1).

Sparsity & CPU | DNN/GRU

Model name	seq_len	#hidden_dim	#coeffs	cpu (hours)
DNN_128_32	128	32	31224	3h52
DNN_128_128	128	128	235128	5h43
DNN_1_32	1	32	31224	3h27
DNN_1_128	1	128	235128	3h40

Table 3: Characteristics and computation times of the built DNN models for different hidden dimensions and seq_len. (cpu times are obtained on a Dell PC, with Intel Core i7 vPRO processor, with 32 Go RAM and NVIDIA RTX A2000 GPU, running on windows 10).



Sparsity & CPU | parsimonious

th	#models (n_m)	#coeffs	cpu (sec)	axis
0.4	2	55	15	1
0.4	3	72	15	2
0.4	4	112	22	3
0.4	3	95	19	4
0.3	3	74	16	1
0.3	5	150	20	2
0.3	7	242	27	3
0.3	5	146	22	4
0.25	4	116	20	1
0.25	6	159	24	2
0.25	5	168	22	3
0.25	9	271	33	4
0.2	6	135	28	1
0.2	9	262	38	2
0.2	13	424	33	3
0.2	13	380	40	4
0.15	13	387	39	1
0.15	10	276	40	2
0.15	16	534	41	3
0.15	20	556	53	4
0.1	22	585	63	1
0.1	29	871	78	2
0.1	19	621	60	3
0.1	32	869	86	4

Table 1: Characteristics (number of polynomials n_m , total number #coeffs of non zero coefficients) and computation times of the identified models for precision thresholds $\text{th} \in \{0.1, 0.15, 0.2, 0.25, 0.3, 0.4\}$. (cpu times are obtained on a MacBook Pro, Apple M3 Pro, 18 Go RAM running on Sequoia 15.3.1).

Sparsity & CPU | DNN/GRU

Model name	seq_len	#hidden_dim	#coeffs	cpu (hours)
DNN_128_32	128	32	31224	3h52
DNN_128_128	128	128	235128	5h43
DNN_1_32	1	32	31224	3h27
DNN_1_128	1	128	235128	3h40

Table 3: Characteristics and computation times of the built DNN models for different hidden dimensions and seq_len. (cpu times are obtained on a Dell PC, with Intel Core i7 vPRO processor, with 32 Go RAM and NVIDIA RTX A2000 GPU, running on windows 10.

Residuals comparison

Axis	Model name	50%	80%	90%	95%	99%
1	DNN_1_32	0.15	0.28	0.36	0.44	0.65
1	DNN_1_128	0.10	0.21	0.28	0.35	0.51
1	DNN_128_32	0.07	0.14	0.19	0.24	0.36
1	DNN_128_128	0.06	0.13	0.17	0.22	0.34
1	Proposed/th=0.20	0.06	0.13	0.19	0.24	0.37
1	Proposed/th=0.15	0.03	0.07	0.11	0.15	0.24
1	Proposed/th=0.1	0.02	0.04	0.06	0.09	0.16
2	DNN_1_32	0.23	0.40	0.53	0.70	1.17
2	DNN_1_128	0.12	0.26	0.37	0.47	0.71
2	DNN_128_32	0.08	0.16	0.23	0.31	0.54
2	DNN_128_128	0.05	0.13	0.17	0.22	0.35
2	Proposed/th=0.20	0.05	0.12	0.17	0.22	0.36
2	Proposed/th=0.15	0.04	0.10	0.14	0.18	0.31
2	Proposed/th=0.1	0.02	0.04	0.06	0.09	0.17
3	DNN_1_32	0.24	0.40	0.48	0.54	0.69
3	DNN_1_128	0.15	0.31	0.40	0.48	0.59
3	DNN_128_32	0.23	0.38	0.47	0.54	0.69
3	DNN_128_128	0.05	0.12	0.16	0.20	0.42
3	Proposed/th=0.20	0.03	0.08	0.11	0.13	0.18
3	Proposed/th=0.15	0.02	0.06	0.08	0.10	0.14
3	Proposed/th=0.1	0.02	0.05	0.07	0.08	0.12
4	DNN_1_32	0.34	0.59	0.71	0.79	0.94
4	DNN_1_128	0.12	0.26	0.40	0.53	0.76
4	DNN_128_32	0.12	0.25	0.35	0.46	0.71
4	DNN_128_128	0.10	0.20	0.28	0.37	0.62
4	Proposed/th=0.20	0.04	0.09	0.12	0.14	0.20
4	Proposed/th=0.15	0.03	0.06	0.09	0.11	0.16
4	Proposed/th=0.1	0.02	0.04	0.06	0.08	0.12

Table 4: Achieved residuals M_q defined by (22), for $q \in \{50\%, 80\%, \dots, 99\% \}$ by the DNN architecture and the proposed piece wise polynomial model on the test experiments.

Alamir, M., & Clavel, S. (2025). An algorithm for sparse piece-wise polynomial residual generation for anomalies detection in industrial manipulator robots. *Control Engineering Practice*, 165, 106545.

Anomaly detector for manipulator robots: Results



Sparsity & CPU | parsimonious

th	#models (n_m)	#coeffs	cpu (sec)	axis
0.4	2	55	15	1
0.4	3	72	15	2
0.4	4	112	22	3
0.4	3	95	19	4
0.3	3	74	16	1
0.3	5	150	20	2
0.3	7	242	27	3
0.3	5	146	22	4
0.25	4	116	20	1
0.25	6	159	24	2
0.25	5	168	22	3
0.25	9	271	33	4
0.2	6	135	28	1
0.2	9	262	38	2
0.2	13	424	33	3
0.2	13	380	40	4
0.15	13	387	39	1
0.15	10	276	40	2
0.15	16	534	41	3
0.15	20	556	53	4
0.1	22	585	63	1
0.1	29	871	78	2
0.1	19	621	60	3
0.1	32	869	86	4

Table 1: Characteristics (number of polynomials n_m , total number #coeffs of non zero coefficients) and computation times of the identified models for precision thresholds $\text{th} \in \{0.1, 0.15, 0.2, 0.25, 0.3, 0.4\}$. (cpu times are obtained on a MacBook Pro, Apple M3 Pro, 18 Go RAM running on Sequoia 15.3.1).

Sparsity & CPU | DNN/GRU

Model name	seq_len	#hidden_dim	#coeffs	cpu (hours)
DNN_128_32	128	32	31224	3h52
DNN_128_128	128	128	235128	5h43
DNN_1_32	1	32	31224	3h27
DNN_1_128	1	128	235128	3h40

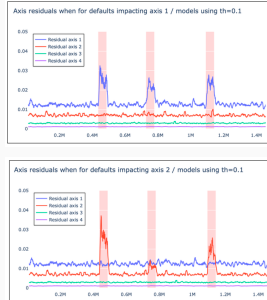
Table 3: Characteristics and computation times of the built DNN models for different hidden dimensions and seq_len. (cpu times are obtained on a Dell PC, with Intel Core i7 vPRO processor, with 32 Go RAM and NVIDIA RTX A2000 GPU, running on windows 10).

Residuals comparison

Axis	Model name	50%	80%	90%	95%	99%
1	DNN_1_32	0.15	0.28	0.36	0.44	0.65
1	DNN_1_128	0.10	0.21	0.28	0.35	0.51
1	DNN_128_32	0.07	0.14	0.19	0.24	0.36
1	DNN_128_128	0.06	0.13	0.17	0.22	0.34
1	Proposed/th=0.20	0.06	0.13	0.19	0.24	0.37
1	Proposed/th=0.15	0.03	0.07	0.11	0.15	0.24
1	Proposed/th=0.1	0.02	0.04	0.06	0.09	0.16
2	DNN_1_32	0.23	0.40	0.53	0.70	1.17
2	DNN_1_128	0.12	0.26	0.37	0.47	0.71
2	DNN_128_32	0.08	0.16	0.23	0.31	0.54
2	DNN_128_128	0.05	0.13	0.17	0.22	0.35
2	Proposed/th=0.20	0.05	0.12	0.17	0.22	0.36
2	Proposed/th=0.15	0.04	0.10	0.14	0.18	0.31
2	Proposed/th=0.1	0.02	0.04	0.06	0.09	0.17
3	DNN_1_32	0.24	0.40	0.48	0.54	0.69
3	DNN_1_128	0.15	0.31	0.40	0.48	0.59
3	DNN_128_32	0.23	0.38	0.47	0.54	0.69
3	DNN_128_128	0.05	0.12	0.16	0.20	0.42
3	Proposed/th=0.20	0.03	0.08	0.11	0.13	0.18
3	Proposed/th=0.15	0.02	0.06	0.08	0.10	0.14
3	Proposed/th=0.1	0.02	0.05	0.07	0.08	0.12
4	DNN_1_32	0.34	0.59	0.71	0.79	0.94
4	DNN_1_128	0.12	0.26	0.40	0.53	0.76
4	DNN_128_32	0.12	0.25	0.35	0.46	0.71
4	DNN_128_128	0.10	0.20	0.28	0.37	0.62
4	Proposed/th=0.20	0.04	0.09	0.12	0.14	0.20
4	Proposed/th=0.15	0.03	0.06	0.09	0.11	0.16
4	Proposed/th=0.1	0.02	0.04	0.06	0.08	0.12

Table 4: Achieved residuals M_q defined by (22), for $q \in \{50\%, 80\%, \dots, 99\%$ by the DNN architecture and the proposed piece wise polynomial model on the test experiments.

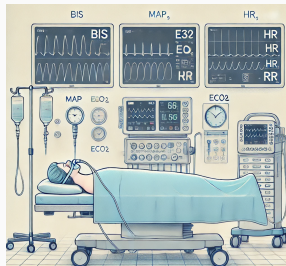
Anomaly detection



Pre-computed thresholds

r	th	axis	η
2	0.1	1	0.016
3	0.1	1	0.015
2	0.3	1	0.071
3	0.3	1	0.066
2	0.1	2	0.010
3	0.1	2	0.009
2	0.3	2	0.035
3	0.3	2	0.033

Alamir, M., & Clavel, S. (2025). An algorithm for sparse piece-wise polynomial residual generation for anomalies detection in industrial manipulator robots. *Control Engineering Practice*, 165, 106545.



Events Detection During Anaesthesia

Events detection during anaesthesia



Patient #1



BIS MAP ETCO2 HR RR

BIS
MAP
ETCO2
HR
RR

Data | Patient #1

•
•
•

Working dataset

Events detection during anaesthesia



Patient #1



Data | Patient #1

Data | Patient #2

Data | Patient #3

Data | Patient #4

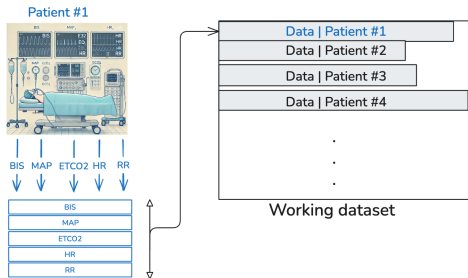
·
·
·

Working dataset

Public dataset

H.-C. Lee, Y. Park, S. B. Yoon, S. M. Yang, D. Park, and C.-W. Jung, "VitalDB, a high-fidelity multi-parameter vital signs database in surgical patients," *Scientific Data*, vol. 9, no. 1, p. 279, Jun. 2022.

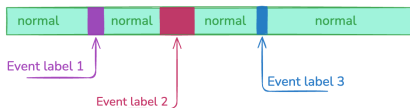
Events detection during anaesthesia



Public dataset

H.-C. Lee, Y. Park, S. B. Yoon, S. M. Yang, D. Park, and C.-W. Jung, "VitalDB, a high-fidelity multi-parameter vital signs database in surgical patients," *Scientific Data*, vol. 9, no. 1, p. 279, Jun. 2022.

Annotating patients recording



Events detection during anaesthesia

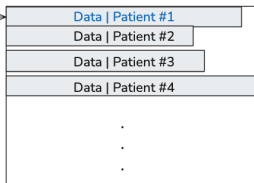


Patient #1



BIS MAP ETCO2 HR RR

BIS
MAP
ETCO2
HR
RR



Working dataset

Public dataset

H.-C. Lee, Y. Park, S. B. Yoon, S. M. Yang, D. Park, and C.-W. Jung, "VitalDB, a high-fidelity multi-parameter vital signs database in surgical patients," *Scientific Data*, vol. 9, no. 1, p. 279, Jun. 2022.

Example of annotation file

	Time_start	Time_end	Label
0	890	1150	intubation
1	1050	1150	pose KTA
2	2500	4000	intubation
3	4500	5000	incision
4	8700	8800	recruitment
5	9400	9500	recruitment
6	11000	11250	recruitment

Annotating patients recording



Event label 1

Event label 2

Event label 3

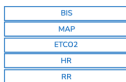
Events detection during anaesthesia



Patient #1



BIS MAP ETCO2 HR RR



Data | Patient #1

Data | Patient #2

Data | Patient #3

Data | Patient #4

Working dataset

Public dataset

H.-C. Lee, Y. Park, S. B. Yoon, S. M. Yang, D. Park, and C.-W. Jung, "VitalDB, a high-fidelity multi-parameter vital signs database in surgical patients," *Scientific Data*, vol. 9, no. 1, p. 279, Jun. 2022.

Example of annotation file

	Time_start	Time_end	Label
0	890	1150	intubation
1	1050	1150	pose KTA
2	2500	4000	intubation
3	4500	5000	incision
4	8700	8800	recruitment
5	9400	9500	recruitment
6	11000	11250	recruitment

Annotating patients recording



Event label 1

Event label 2

Event label 3

Label	#Instances	#Patients	#Patients_train
normal	213878	19	6
pain	8765	6	1
incision	7003	16	6
noception	4911	7	1
intubation	4420	10	3
recruitment	3507	10	3
pose KTA	2760	15	6
Hypovolemia / vasoplegia	2553	3	1
Excess/default sedation	1763	2	0
artefact	1302	2	0
revel	999	1	0
induction	874	4	0
error_BIS	594	1	0
reaction vagale ?	501	1	0
Breuncho-spasm	501	1	0
stop_GA	450	1	0
exclusion pulmonaire	401	1	1
error_acq_HIR	201	1	0
sampling	162	2	0
action_chir_posumom	151	1	0
exhalation	132	3	0
control_unipulm	121	1	0

Events detection during anaesthesia



Patient #1



BIS MAP ETCO2 HR RR

BIS
MAP
ETCO2
HR
RR

Data Patient #1
Data Patient #2
Data Patient #3
Data Patient #4
.
.
.

Working dataset

19 annotated patients
5 used for training

Annotating patients recording



Event label 1

Event label 2

Event label 3

Public dataset

H.-C. Lee, Y. Park, S. B. Yoon, S. M. Yang, D. Park, and C.-W. Jung, "VitalDB, a high-fidelity multi-parameter vital signs database in surgical patients," *Scientific Data*, vol. 9, no. 1, p. 279, Jun. 2022.

Example of annotation file

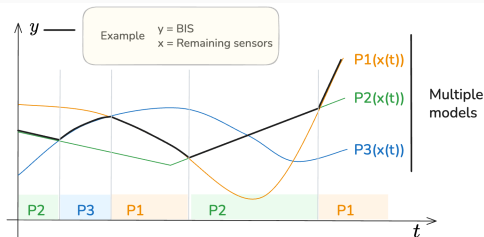
	Time_start	Time_end	Label
0	890	1150	intubation
1	1050	1150	pose KTA
2	2500	4000	intubation
3	4500	5000	incision
4	8700	8800	recruitment
5	9400	9500	recruitment
6	11000	11250	recruitment

Label	#Instants	#Patients	#Patients_train
normal	213878	19	6
pain	8765	6	1
incision	7003	16	6
nociption	4911	7	1
intubation	4420	10	3
recruitment	3507	10	3
pose KTA	2760	15	6
Hypovolemia / vasoplegia	2453	3	1
Excess/default sedation	1763	2	0
artefact	1302	2	0
reveil	999	1	0
induction	874	4	0
error_BIS	594	1	0
réaction vagale ?	501	1	0
Broncho-spasm	501	1	0
stop_GA	450	1	0
exclusion pulmonaire	401	1	1
error_acq_HR	201	1	0
sampling	162	2	0
action_chir_poumon	151	1	0
extubation	132	3	0
control_unipulm	121	1	0

Events detection during anaesthesia: Methodology

Five piece-wise polynomial relationships

Each corresponds to a particular choice of the pair (y, x)



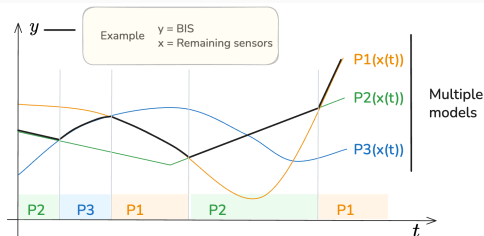
Find a set of models such that the following invariant implicit relationship is satisfied

$$e(x, y) = \min_{k=1}^3 |y - P_k(x)| \approx 0$$

Events detection during anaesthesia: Methodology

Five piece-wise polynomial relationships

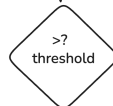
Each corresponds to a particular choice of the pair (y, x)



Find a set of models such that the following invariant implicit relationship is satisfied

$$e(x, y) = \min_{k=1}^3 |y - P_k(x)| \approx 0$$

design a threshold



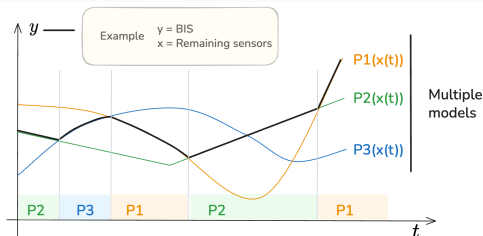
Event occurred

No event

Events detection during anaesthesia: Methodology

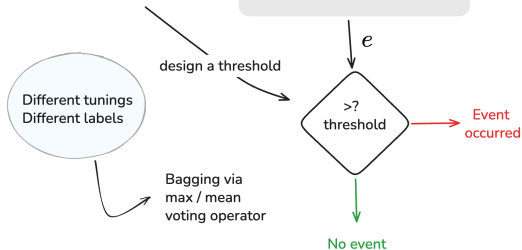
Five piece-wise polynomial relationships

Each corresponds to a particular choice of the pair (y, x)



Find a set of models such that the following invariant implicit relationship is satisfied

$$e(x, y) = \min_{k=1}^3 |y - P_k(x)| \approx 0$$



Sparsity

Hyper parameter of the piece-wise relationships

th	0.05	0.12	0.10	0.17	0.15	0.20	0.25	0.3
BIS	(68, 484)	(14, 123)	(22, 150)	(7, 65)	(12, 117)	(6, 42)	(7, 66)	(4, 53)
MAP	(11, 115)	(6, 54)	(7, 37)	(4, 43)	(4, 31)	(3, 28)	(3, 29)	(2, 31)
HR	(22, 207)	(8, 72)	(10, 95)	(4, 54)	(7, 70)	(4, 50)	(4, 33)	(3, 25)
RR	(24, 151)	(7, 48)	(8, 74)	(5, 60)	(7, 66)	(4, 37)	(4, 30)	(3, 24)
ETCO2	(26, 163)	(8, 86)	(8, 70)	(6, 62)	(6, 51)	(4, 40)	(3, 42)	(4, 51)

(#polynomials, #monomials)

Choice of the label

Events detection during anaesthesia: Results

Sparsity

Hyper parameter of the piece-wise relationships

t.h	0.05	0.12	0.10	0.17	0.15	0.20	0.25	0.3
BIS	(68, 484)	(14, 123)	(22, 150)	(7, 65)	(12, 117)	(6, 42)	(7, 66)	(4, 53)
MAP	(11, 115)	(6, 54)	(7, 37)	(4, 43)	(4, 31)	(3, 28)	(3, 29)	(2, 31)
HR	(22, 207)	(8, 72)	(10, 95)	(4, 54)	(7, 70)	(4, 50)	(4, 33)	(3, 25)
RR	(24, 151)	(7, 48)	(8, 74)	(5, 60)	(7, 66)	(4, 37)	(4, 30)	(3, 24)
ETCO2	(26, 163)	(8, 86)	(8, 70)	(6, 62)	(6, 51)	(4, 40)	(3, 42)	(4, 51)

(#polynomials, #monomials)

Choice of the label

Detection

Detection Rate — Averaging window of 2min

$n_w = 120$ max-aggregation	FA=0.05	FA=0.1	FA=0.15	FA=0.2	FA=0.25	FA=0.3	#Patients	#Partient_train
extubation	100	100	100	100	100	100	3	0
control_unipulm	100	100	100	100	100	100	1	0
induction	28	57	85	100	100	100	4	0
artefact	40	50	70	80	90	90	2	0
action_chir_poumon	0	0	100	100	100	100	1	0
intubation	22	38	66	75	77	80	10	3
pose KTA	4	31	72	72	86	86	15	6
réaction vagale ?	0	0	50	75	75	75	1	0
stop_GA	0	33	33	33	66	100	1	0
recruitment	10	27	48	58	58	58	10	3
Excess/default sedation	7	7	21	64	78	78	2	0
error_acq_HR	0	0	0	0	100	100	1	0
pain	1	13	34	45	46	49	6	1
reveil	12	25	37	37	37	37	1	0
incision	0	5	18	27	37	39	16	6
normal	5	9	15	19	25	29	19	6
nociception	5	5	12	17	30	32	7	1

Events detection during anaesthesia: Results

Sparsity

Hyper parameter of the piece-wise relationships

th	0.05	0.12	0.10	0.17	0.15	0.20	0.25	0.3
BIS	(68, 484)	(14, 123)	(22, 150)	(7, 65)	(12, 117)	(6, 42)	(7, 66)	(4, 53)
MAP	(11, 115)	(6, 54)	(7, 37)	(4, 43)	(4, 31)	(3, 28)	(3, 29)	(2, 31)
HR	(22, 207)	(8, 72)	(10, 95)	(4, 54)	(7, 70)	(4, 50)	(4, 33)	(3, 25)
RR	(24, 151)	(7, 48)	(8, 74)	(5, 60)	(7, 66)	(4, 37)	(4, 30)	(3, 24)
ETCO2	(26, 163)	(8, 86)	(8, 70)	(6, 62)	(6, 51)	(4, 40)	(3, 42)	(4, 51)

(#polynomials, #monomials)

Choice of the label

Detection

Detection Rate — Averaging window of 1min

$n_w = 60$ max-aggregation	FA=0.05	FA=0.1	FA=0.15	FA=0.2	FA=0.25	FA=0.3	#Patients	#Partient_train
extubation	100	100	100	100	100	100	3	0
control_unipulm	100	100	100	100	100	100	1	0
induction	14	50	85	85	85	85	4	0
flush_art	0	0	100	100	100	100	1	0
artefact	33	57	66	76	76	85	2	0
pose_KTA	11	44	57	75	77	82	15	6
intubation	17	38	61	67	73	76	10	3
stop_GA	14	28	42	42	71	85	1	0
réaction vagale ?	0	25	50	62	62	75	1	0
Excess/default sedation	6	10	37	65	65	82	2	0
recruitment	8	34	44	50	53	55	10	3
deconnexion_respi	0	0	0	0	100	100	1	0
action_chir_poumon	0	0	50	50	50	50	1	0
pain	0	23	30	38	43	50	6	1
error_acq_HR	0	0	0	33	66	66	1	0
reveil	18	18	25	31	31	31	1	0
incision	0	5	15	26	32	40	16	6
normal	4	10	15	20	25	29	19	6
nociception	2	6	11	20	27	28	7	1

Events detection during anaesthesia: Results

Sparsity

Hyper parameter of the piece-wise relationships

th	0.05	0.12	0.10	0.17	0.15	0.20	0.25	0.3
BIS	(68, 484)	(14, 123)	(22, 150)	(7, 65)	(12, 117)	(6, 42)	(7, 66)	(4, 53)
MAP	(11, 115)	(6, 54)	(7, 37)	(4, 43)	(4, 31)	(3, 28)	(3, 29)	(2, 31)
HR	(22, 207)	(8, 72)	(10, 95)	(4, 54)	(7, 70)	(4, 50)	(4, 33)	(3, 25)
RR	(24, 151)	(7, 48)	(8, 74)	(5, 60)	(7, 66)	(4, 37)	(4, 30)	(3, 24)
ETCO2	(26, 163)	(8, 86)	(8, 70)	(6, 62)	(6, 51)	(4, 40)	(3, 42)	(4, 51)

(#polynomials, #monomials)

Choice of the label

Detection

Detection Rate — Averaging window of 30 sec

$n_w = 10 \text{max-aggregation}$	FA=0.05	FA=0.1	FA=0.15	FA=0.2	FA=0.25	FA=0.3	#Patients	#Patient_train
pose_ABP	100	100	100	100	100	100	1	0
end_procedure	100	100	100	100	100	100	1	0
extubation	84	92	92	100	100	100	3	0
vent_leak	20	100	100	100	100	100	1	0
art_disconnect	0	100	100	100	100	100	1	0
control_unipulm	50	75	75	75	91	91	1	0
induction	13	51	79	79	80	82	4	0
début coelioscopie	0	40	80	80	80	80	1	0
artefact	18	50	57	70	74	82	2	0
pose KTA	7	33	49	62	67	73	15	6
flush_art	0	37	50	62	62	75	1	0
intubation	13	37	48	54	58	62	10	3
Excess/default sedation	1	18	46	59	65	71	2	0
deconnexion_respi	0	0	0	57	100	100	1	0
stop_GA	11	27	34	50	61	72	1	0
réaction vagale ?	0	24	42	58	60	66	1	0
pain	2	18	27	37	45	51	6	1
recruitment	6	20	29	34	38	43	10	3
error_acq_HR	0	0	5	25	45	80	1	0
reveil	11	17	22	25	27	30	1	0
normal	4	10	14	20	24	29	19	6
incision	1	6	14	20	26	31	16	6
nociception	1	5	9	15	18	21	7	1
action_chir_poumon	0	13	13	13	13	13	1	0
error_BIS	0	0	3	5	5	27	1	0
sampling	0	0	0	0	18	18	2	0



Parsimonious piece-wise polynomial models in Robotics



Inverse kinematic problems

$$\text{Torque}_i \approx \text{ML}(\underbrace{q, \dot{q}, \ddot{q}}_{x \in \mathbb{R}^{18}}) \quad i = 1, \dots, 6$$

Working data

6-axis robot | **380** trajectories |
Average duration 51 sec |
250 Hz | **4,858,087** samples.

Splitting the data

1/10 samples in the first 5% for training
→ **24,290** samples
→ **4,615,183** samples for test.

Comparison to $\neq$ DNNs structures

MLP | CNN | Transformer

Xpwpol

$$\hat{T} \approx \frac{1}{n_V} \sum_{\ell=1}^{n_V} P_{\kappa^*(x)}(x)$$

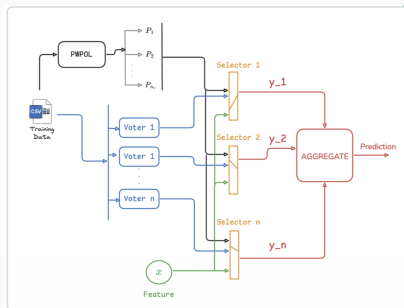


Table 1: Use-case 1: Number of polynomials and number of voters for the explicit piece-wise polynomial models

Axis	1	2	3	4	5	6
n_m	15	2	4	21	25	25
nVoters	4	1	3	4	2	6

Table 4: Use-case 1: Results on the test dataset: Normalized residuals (nMAE, nMSE) | Large DNN models

Axis	MLP (Large)	CNN (Large)	Transformer (Large)	xwpol
1	(0.23, 0.06)	(0.22, 0.05)	(0.20, 0.05)	((0.22, 0.06))
2	(0.11, 0.02)	(0.09, 0.01)	(0.09, 0.01)	(0.12, 0.02)
3	(0.19, 0.05)	(0.15, 0.03)	(0.15, 0.03)	(0.20, 0.05)
4	(0.26, 0.09)	(0.24, 0.07)	(0.22, 0.06)	(0.37, 0.17)
5	(0.53, 0.30)	(0.40, 0.16)	(0.39, 0.17)	(0.60, 0.33)
6	(0.37, 0.17)	(0.28, 0.11)	(0.31, 0.12)	(0.52, 0.31)

Table 6: Use-case 1: Number of active coefficients.

Axis	MLP	CNN	Transformer	xwpol
1				387
2				66
3	Small: 14,086	Small: 10,542	Small: 11,190	183
4	Large: 617,990	Large: 618,598	Large: 797,958	464
5				431
6				688

Alamir, M., & Clavel, S. Identifying Explicit Parsimonious Piece-wise Polynomial Relationships in Industrial time-series: Application to manipulator robots. *Control Engineering Practice* (submitted, 2026).

Table 7: Use-case 1: Computation times for model fitting.

Axis	MLP	CNN	Transformer	xwpol
1				106 sec
2		(pytorch-GPU)		35 sec
3				28 sec
4		> 2h		142 sec
5				162 sec
6		See Section 4.1.4		173 sec

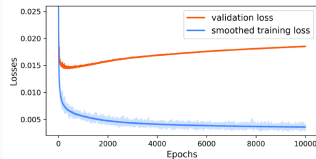


Table 5: Use-case 1: Centred Generalization Gap α_c , defined by (21) | Large DNN models

Axis	MLP (Large)	CNN (Large)	Transformer (Large)	xwpol
1	0.64	0.57	0.33	0.00
2	1.20	1.25	0.80	0.01
3	1.71	1.50	1.14	0.02
4	1.00	1.00	0.69	0.11
5	2.31	1.86	1.44	0.03
6	1.64	1.55	1.38	0.09

$$\alpha_c := \left| \frac{\text{nMAE}_{\text{test}}}{\text{nMAE}_{\text{train}}} - 1 \right|$$



Sparsity & generalization to unseen contexts



Inverse kinematic problems

$$\text{Torque}_i \approx \text{ML}(\underbrace{q, \dot{q}, \ddot{q}}_{x \in \mathbb{R}^{12}}) \quad i = 1, \dots, 4$$

Working data

4-axis robot | **835** trajectories |

Context 1: Small excursions | (635)

Context 2: Large excursions | (200)

250 Hz | **3,246,892** samples (small=large)

Splitting the data

Training 24,360 samples (context 1)

→ Test-small **1,599,596** (context 1)

→ Test-large **1,622,936** (context 2)

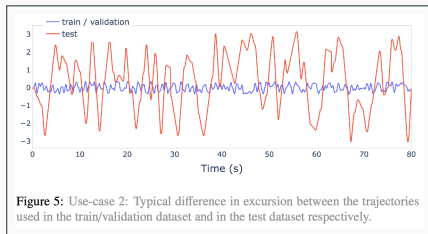
Comparison to $\neq$ DNNs structures

Transformer (Best DNN!)

Xpwpol

$$\hat{T} \approx \frac{1}{n_V} \sum_{\ell=1}^{n_V} P_{\kappa^*}(x)(x)$$

The two contexts (different excursions)





Inverse kinematic problems

$$\text{Torque}_i \approx \text{ML}(\underbrace{q, \dot{q}, \ddot{q}}_{x \in \mathbb{R}^{12}}) \quad i = 1, \dots, 4$$

Working data

4-axis robot | **835** trajectories |

Context 1: Small excursions | (635)

Context 2: Large excursions | (200)

250 Hz | **3,246,892** samples (small=large)

Splitting the data

Training 24,360 samples (context 1)

→ Test-small **1,599,596** (context 1)

→ Test-large **1,622,936** (context 2)

Comparison to $\neq$ DNNs structures

Transformer (Best DNN!)

XpwPol

$$\hat{T} \approx \frac{1}{n_V} \sum_{\ell=1}^{n_V} P_{\kappa^*}(x)(x)$$

Piece-wise polynomial models

Table 8: Use-case 2: Number of polynomials, number of voters and total number of active coefficients for the explicit piece-wise polynomial models. For the number of coefficients involved in the DNN models, see Table 6

Axis	1	2	3	4
n_m	4	5	5	8
nVoters	1	1	1	1
# coefficients	273	104	194	282

Alamir, M., & Clavel, S. Identifying Explicit Parsimonious Piece-wise Polynomial Relationships in Industrial time-series: Application to manipulator robots. **Control Engineering Practice** (submitted, 2026).



Inverse kinematic problems

$$\text{Torque}_i \approx \text{ML}(\underbrace{q, \dot{q}, \ddot{q}}_{x \in \mathbb{R}^{12}}) \quad i = 1, \dots, 4$$

Working data

4-axis robot | **835** trajectories |

Context 1: Small excursions | (635)

Context 2: Large excursions | (200)

250 Hz | **3,246,892** samples (small=large)

Splitting the data

Training 24,360 samples (context 1)

→ Test-small **1,599,596** (context 1)

→ Test-large **1,622,936** (context 2)

Comparison to $\neq$ DNNs structures

Transformer (Best DNN!)

Xpwpol

$$\hat{T} \approx \frac{1}{n_V} \sum_{\ell=1}^{n_V} P_{\kappa^*}(x)(x)$$

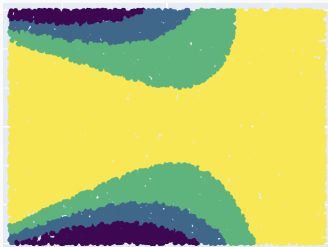
Results on test datasets for the two contexts

Table 9: Use-case 2: Normalized residuals (nMAE, nMSE) | Validation

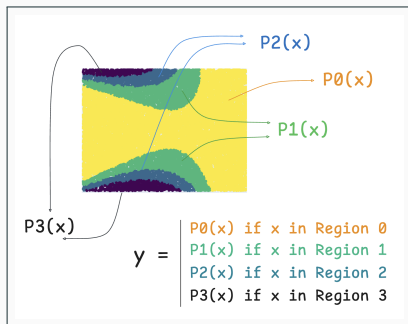
Axis	Transformer (Validation)	xpwpol (Validation)
1	(0.08, 0.01)	(0.10, 0.01)
2	(0.10, 0.01)	(0.12, 0.02)
3	(0.13, 0.03)	(0.10, 0.01)
4	(0.24, 0.10)	(0.20, 0.05)

Table 10: Use-case 2: Normalized residuals (nMAE, nMSE) | Test

Axis	Transformer (Test)	xpwpol (Test)
1	(1.38, 1.53)	(0.36, 0.13)
2	(1.73, 2.57)	(0.28, 0.08)
3	(0.77, 0.63)	(0.37, 0.14)
4	(1.96, 3.79)	(2.33, 5.68)



Fault Detection under **Multiple Unknown Contexts**



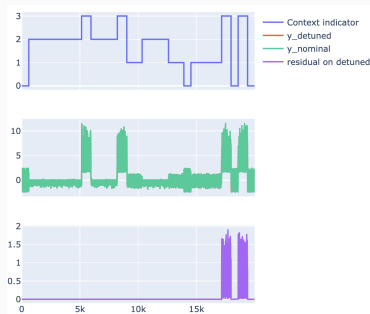
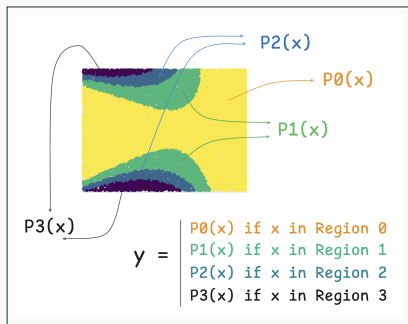
Problem with: features ($n_x = 2$) | 4 contexts

Random generation: regions & polynomials P_i .

Fault: detune the coefficients

- in a single context
- in a short interval of the 1/2 half of dataset

<https://doi.org/10.34740/kaggle/dsv/13462796>



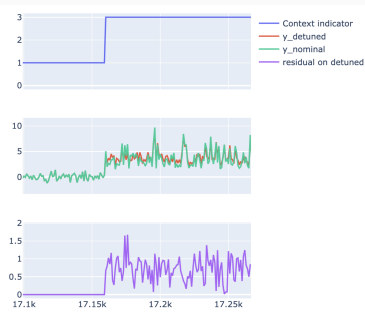
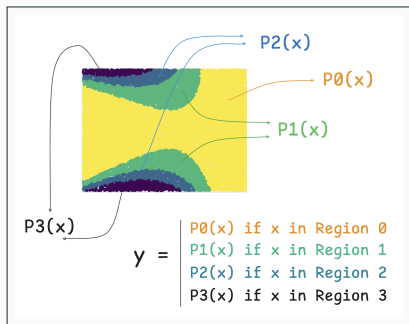
Problem with: features ($n_x = 2$) | 4 contexts

Random generation: regions & polynomials P_i .

Fault: detune the coefficients

- in a single context
- in a short interval of the 1/2 half of dataset

<https://doi.org/10.34740/kaggle/dsv/13462796>



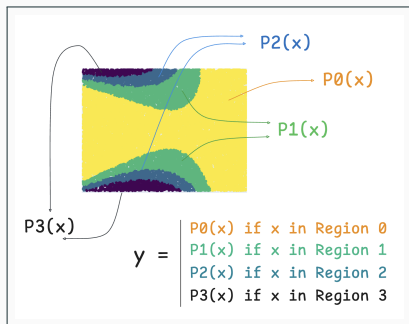
Problem with: features ($n_x = 2$) | 4 contexts

Random generation: regions & polynomials P_i .

Fault: detune the coefficients

- in a single context
- in a short interval of the 1/2 half of dataset

<https://doi.org/10.34740/kaggle/dsv/13462796>



Problem with: features ($n_x = 2$) | 4 contexts

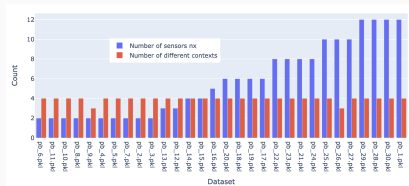
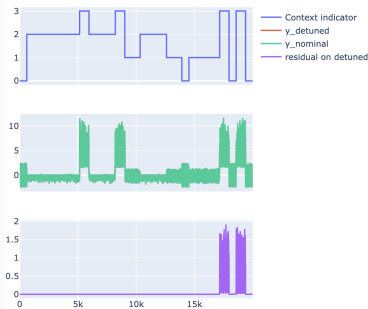
Random generation: regions & polynomials P_i .

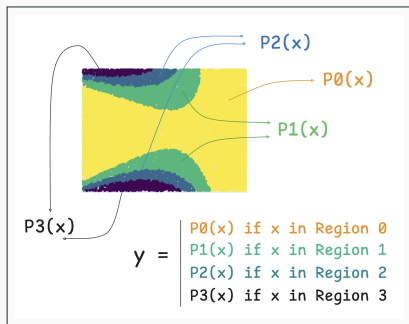
Fault: detune the coefficients

- in a single context
- in a short interval of the 1/2 half of dataset

30 random instances

<https://doi.org/10.34740/kaggle/dsv/13462796>





Problem with: features ($n_x = 2$) | 4 contexts

Random generation: regions & polynomials P_i .

Fault: detune the coefficients

- in a single context
- in a short interval of the 1/2 half of dataset

30 random instances | Raw/Normalized.

<https://doi.org/10.34740/kaggle/dsv/13462796>

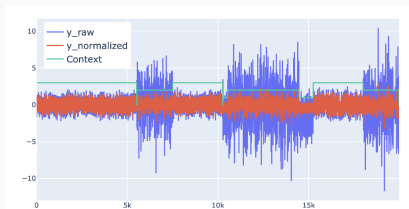
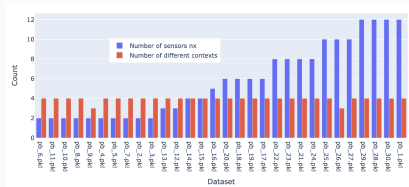


Fig. 8. Example of normalization performed on the file pb_28.pkl included in the benchmark dataset.



- ✓ Train a model on the first half ^a
- ✓ Test on the remaining half ^b

Residual for a given hyper-parameter

$$e_{th}(x, y) := \min_{\kappa=1}^{n_r(th)} |y - P_{\kappa}^{(th)}(x)|$$

Voting over several settings^c

$$e(x, y) := \frac{1}{\text{card}(\mathcal{T}_h)} \sum_{th \in \mathcal{T}_h} e_{th}(x, y)$$

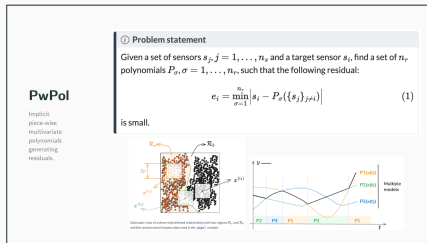
Average prediction over a window

$$\bar{e}(k) := \frac{1}{n_w} \left[\sum_{t=k-n_w+1}^{t=k} e(t) \right]$$

^aFault-free

^bIncludes a faulty interval on a single context.

^c $\mathcal{T}_h := \{0.15, 0.2, 0.25, 0.3\}$



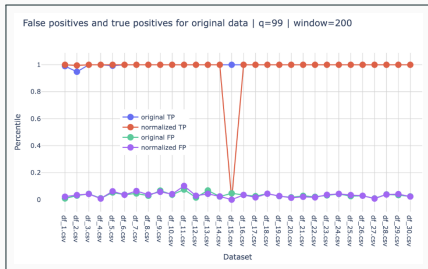
Alarm threshold definition

$$\bar{e}_k \geq \text{percentile}(\bar{e}|\text{train}, 99)$$

Performance metrics

(False Alarm, Detection Rate)

Results after aggregation (Bagging)



- ✓ Detection rate $\approx$ perfect except for experiment # 15
- ✓ Failure only on the normalized version of exp. # 15
- ✓ Bagging is quite effective.

Results of the different voters

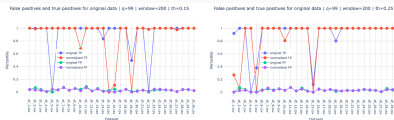
$$th \in \{0.15, 0.2, 0.25, 0.3\}$$


Fig. 10. Detection and false alarm rates of the solution associated to the precision threshold $th = 0.15$ over the 30 files of the benchmark dataset.

Fig. 12. Detection and false alarm rates of the solution associated to the precision threshold $th = 0.25$ over the 30 files of the benchmark dataset.



Fig. 11. Detection and false alarm rates of the solution associated to the precision threshold $th = 0.2$ over the 30 files of the benchmark dataset.

Fig. 13. Detection and false alarm rates of the solution associated to the precision threshold $th = 0.3$ over the 30 files of the benchmark dataset.

Alamir, M. A benchmark for context-dependent parametric anomaly detection in industrial time series.
<https://cnrs.hal.science/hal-05533907>.

 **MizoPol** [Home](#) [Overview](#) [Plars](#) [g2sys](#) [pwpol](#) [xwpol](#) [rlars](#) [Graph](#) [Digital twins](#) [About](#) [Mazen Alami](#)  

MizoPol

Preliminaries

Overview

Polynomials

piecewise polynomials

Parsimony: Intro

Parsimony: Example 1

Parsimony: Example 2

True/False anomalies

plars

plars: Introduction

plars: Principle

plars: Example of use

plars: GUI

plars: Robotic

g2sys

g2sys: Introduction

g2sys: Principle

g2sys: Delays matter

g2sys: Hydraulic rig

g2sys: Compressor

g2sys: Valves

g2sys: turbofan

MizoPol

A set of tools for identification, normality characterization and anomaly detection in industrial time-series.



MizoPol is a set of python modules¹ that addresses the problem of **identifying (piece-wise) polynomial relationships** between time series. This identification is detrimental in solving the following problems:

- Anomaly detection by normality characterization
- Understanding the coupling between sensors beyond linear correlations
- Identifying dynamics governing nonlinear unmodelled systems
- Creating virtual sensors
- Generating parametric subtil defaults to auto-train models
- help designing digital twins

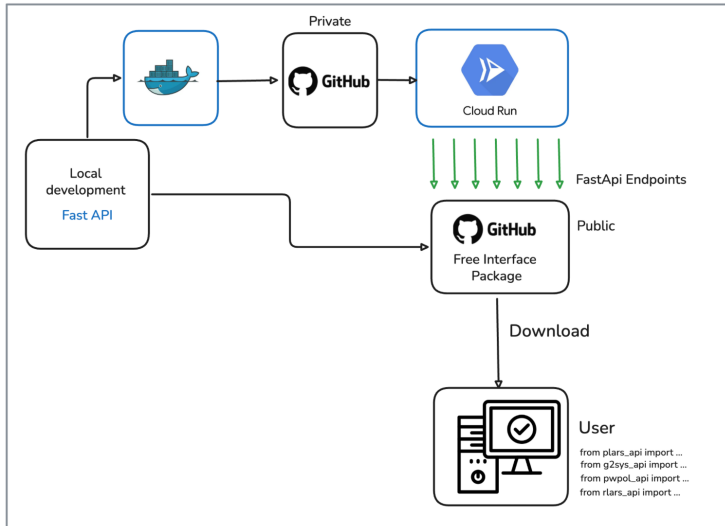
 Domains of application

Achieving the above listed tasks paves the way towards addressing many important domains of interest in industry such as:

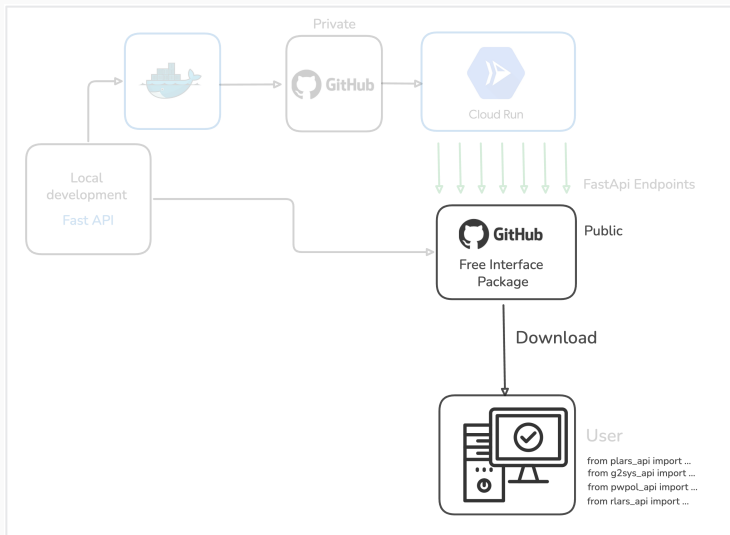
https://mazenalamir.github.io/mizopol_documentation/

Deployment | API

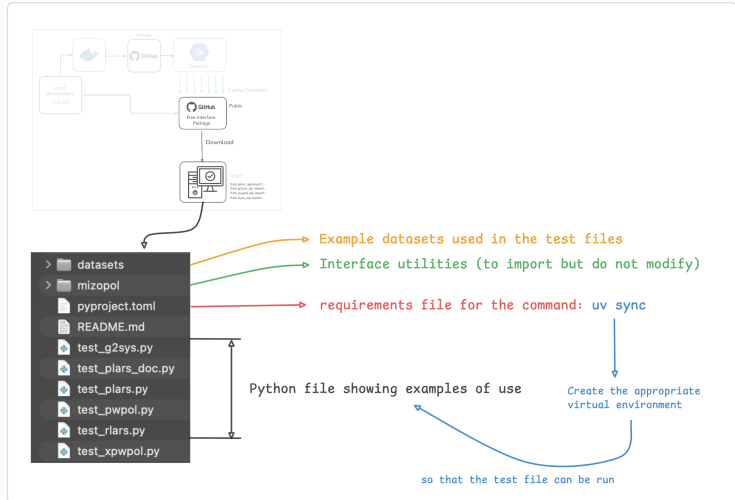
Deployment and API



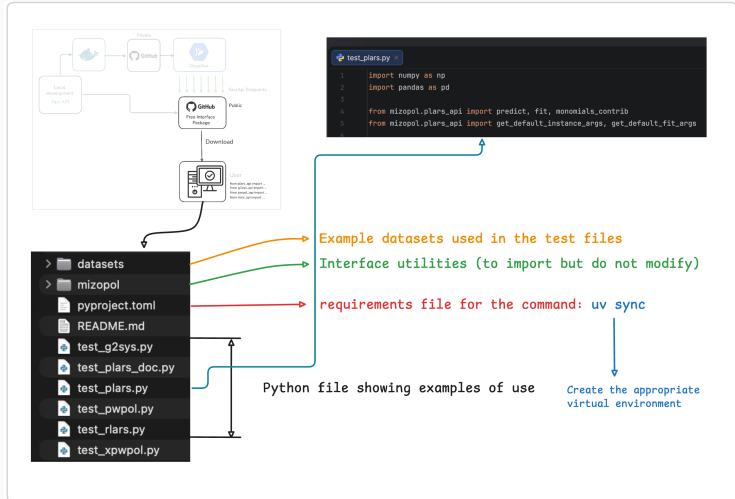
Deployment and API



Deployment and API



Deployment and API



 **MizoPol** [Home](#) [Overview](#) [Plars](#) [g2sys](#) [pwpol](#) [xwpol](#) [rlars](#) [Graph](#) [Digital twins](#) [About](#) [Mazen Alami](#)  

MizoPol

Preliminaries

Overview

Polynomials

piewe-wise polynomials

Parsimony: Intro

Parsimony: Example 1

Parsimony: Example 2

True/False anomalies

plars

plars: Introduction

plars: Principle

plars: Example of use

plars: GUI

plars: Robotic

g2sys

g2sys: Introduction

g2sys: Principle

g2sys: Delays matter

g2sys: Hydraulic rig

g2sys: Compressor

g2sys: Valves

g2sys: turbofan

MizoPol

A set of tools for identification, normality characterization and anomaly detection in industrial time-series.



MizoPol is a set of python modules¹ that addresses the problem of **identifying (piece-wise) polynomial relationships** between time series. This identification is detrimental in solving the following problems:

- Anomaly detection by normality characterization
- Understanding the coupling between sensors beyond linear correlations
- Identifying dynamics governing nonlinear unmodelled systems
- Creating virtual sensors
- Generating parametric subtil defaults to auto-train models
- help designing digital twins

 Domains of application

Achieving the above listed tasks paves the way towards addressing many important domains of interest in industry such as:

https://mazenalamir.github.io/mizopol_documentation/